BRINGING THE GRAIL TO THE CCRMA STAGE

Fernando Lopez-Lezcano
CCRMA
Stanford University, USA
nando@ccrma.stanford.edu

Christopher Jette
CCRMA
Stanford University, USA
jette@ccrma.stanford.edu

ABSTRACT

The Stage, a small concert hall at CCRMA, Stanford University, was designed as a multi-purpose space when The Knoll, the building that houses CCRMA, was renovated in 2003/5. It is used for concerts, installations, classes and lectures, and as such it needs to be always available and accessible. Its support for sound diffusion evolved from an original array of 8 speakers in 2005, to 16 speakers in a 3D configuration in 2011, with several changes in speaker placement over the years that optimized the ability to diffuse pieces in full 3D surround. This paper describes the evolution of the design and a significant upgrade in 2017 that made it capable of rendering HOA (High Order Ambisonics) of up to 5th or 6th order, without changing the ease of operation of the existing design for classes and lectures, and making it easy for composers and concert presenters to work with both the HOA and legacy 16 channel systems.

1. INTRODUCTION

We have been hosting concerts at CCRMA since it was created in the 70's. In 2009 we started expanding our concert diffusion capabilities while gearing up for the inaugural season of a new concert hall being built at Stanford, the Bing Concert Hall. In 2013 we were able to use our newly created GRAIL system (the Giant Radial Array for Immersive Listening) to diffuse concerts with out own "portable" speaker array with up to 24 speakers and 8 subwoofers arranged in a dome configuration for full 3D surround sound diffusion [1].

Figure 1: CCRMA Concert in the Bing Studio with the GRAIL

By 2011 our Listening Room Studio included a 22.4 speaker array in a full 3D configuration (with speakers below an acoustically transparent grid floor), which could accurately decode periphonic (full 3D) 3rd order Ambisonics. Our upgraded GRAIL concert diffusion system was also able to render up to 3rd order Ambisonics, or even 4th order if some errors in rendering were ignored. This was made possible by the publication of algorithms that allowed the design of HOA decoders for irregular arrays [2]. In particular, the release of the Ambisonics Decoder Toolkit software package written by Aaron Heller [3][4], which included software implementations of the aforementioned research, simplified the task of designing decoders. This work enabled the creation of successful diffusion strategies for irregular speaker placement in the Bing Concert Hall and its rehearsal space (the Studio), as well as other spaces. Both systems benefited from an open architecture based on the GNU/Linux operating system and many free audio software packages that, combined, allowed us to tailor the system to our specific needs.

We have curated many concerts with content of varied spatial resolution. As composers went on to create works requiring more speakers for a higher Ambisonic order decode, the limitations of our systems became apparent. While Ambisonics is well known for a graceful degradation of the spatial resolution when not enough speakers are available for the original order of the piece, the state of research and artistic creation was moving towards orders that were higher than what we could support.

1.1. From WFS tests to HOA in the Stage

In 2011 we bought 32 small speakers (Adam A3X) to create an experimental WFS array. Over the next few years we used it for demos and classes, but other than a couple of concert performances the system was used very sparingly. On the other hand, our Stage concert hall had a complement of 16 speakers and 8 subwoofers, which limited our ability to render full 3D HOA (we had been recently using a 32.8 system for our off-site concerts).

In an effort to upgrade our dedicated diffusion space at CCRMA, we proposed to re-purpose the "unused" speakers and add them to the existing Stage diffusion system. This addition would increase the total count of speakers to 48, and preliminary studies determined that we would be able to render up to 6th order Ambisonics quite accurately. Natasha Barret's research [5] points to diminishing returns in spatial performance for 7th and higher order decoding, so we felt confident that moving to a fifth or sixth order system would be adequate for our needs and a worthwhile upgrade.

The design and implementation of this upgrade ended up being anything but easy.

2. REQUIREMENTS

The existing system in the Stage consisted of 8 movable tower stands, each one housing a main speaker (four S3A and four P33 Adam high quality mid-field studio monitors) and a subwoofer (M-Audio

SBX10). In addition to those, we had 8 Adam P22 speakers hanging from the trusses and arranged as a ring of 6 with an additional two more overhead. All 16 speakers could be individually addressed from a Yamaha DM1000 mixer, with some limitations as the subwoofers were paired to the 8 main speakers - we used their internal crossovers - and could not be used by the upper 8 speakers.

The Stage is not only a concert hall, it is also regularly used for classes, lectures, demos and other events that do not need or want a high spatial resolution speaker array. In fact, the majority of users require access to just stereo playback. As the CCRMA concert events combine live performers, touring musicians and researchers, many concerts do not deal with 3D surround sound and use mostly stereo projection. The existing flexible 16 channel system allowed for creative diffusion using a combination of speakers and provided flexibility in which orientation the space could be used.

One of the key requirements for the upgrade was that the existing system and methods of operation would not be changed. Furthermore, the space sometimes is used to accommodate big audiences (for its size), so any addition to the Stage could not permanently encroach in the floor space available for setting up chairs for events.

These varied requirements complicated the design process in ways which we had not anticipated.

We were required to:

1. have a mode of operation that would keep the existing design, 8 main speaker and subwoofer towers plus 8 secondary speakers hanging from the ceiling trusses, all of them driven directly from our DM1000 digital mixer
2. not degrade the performance of the existing system in any way, including the low latency achievable with the digital mixer, appropriate for live performances
3. have a way to easily switch from the basic system to a fully expanded speaker array which added 32 speakers, all of them controlled through a single Linux based computer similar to the one managing diffusion tasks in our Listening Room [6][7]
4. have the ability to physically move the additional small speakers positioned at ear level out of the way, so that they would not interfere with the existing floor footprint of the diffusion system
5. easily switch between the two modes of operation, preferably with "one big switch" that would need no expertise from the operator
6. the system had to be "low cost"

This created a situation with many mutually incompatible system requirements from a design standpoint.

3. FEASABILITY TESTING

Before starting the upgrade a practical question had to be answered: were the tiny A3X speakers good enough (in quantity) to be able to produce enough SPL for a concert diffusion situation? Matt Wright and Christopher Jette organized a quick test session in which we installed 16 speakers in a ring at ear level (on top of chairs and plastic bins!) and drove them from our GRAIL concert control computer. This test was successful and confirmed that they were up to the task, but only if properly equalized, so we could go ahead with the upgrade.

4. LOCATION, LOCATION, LOCATION

Where and how to mount all speakers was a difficult task, made harder by the rectangular shape of the room and the presence of trusses that hold the cathedral-style ceiling. To arrive at a preliminary even distribution in space we used a simple successive approximation software that treats speaker locations as electrons that repel each other, and determines the approximate ideal locations of the speakers [8]. Additional constraints were introduced in the software to "fix" the position of the existing 16 speakers in space (remember that our design must be a superset of the existing system), and see where the rest of the speakers would fall.

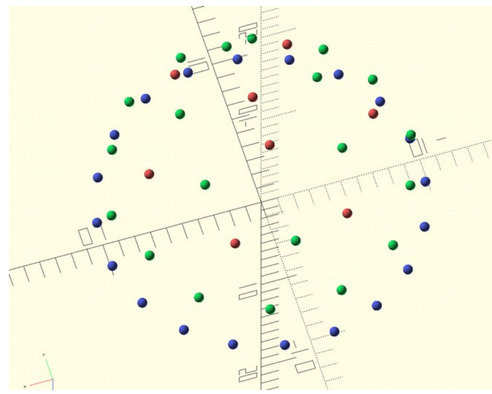

Figure 2: Ideal projection of speaker locations on a hemisphere (red dots: original upper 8 speakers, blue dots: ear level speakers)

A simple geometrical model of the Stage created in OpenSCAD [9] was used to project those ideal locations into the walls and ceiling of the Stage, to see where we might approximate the ideal locations in space with real mounting points. It was challenging to find locations which would not be shadowed by the ceiling trusses for most of the audience seating space, and in a couple of instances there was unavoidable shadowing that we had to ignore.

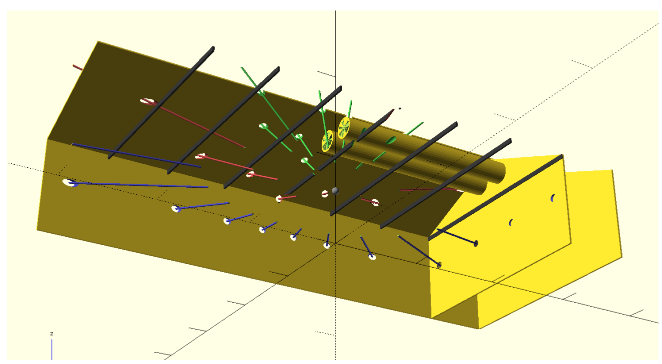

Figure 3: OpenSCAD model of the Stage (seen from below) with speaker location projections, the cylinders partially represent the A/C ducts, the black beams are the lower part of the trusses

We used ADT (the Ambisonics Decoder Toolkit)[3][4] as a design verification tool, in particular the energy and particle velocity graphs helped us determine if the proposed mounting locations for the speakers would provide uniform coverage for the desired Ambisonics orders (5ht and 6th order was the goal). Other diffusion

methods (VBAP, etc) would also benefit from a uniform spatial distribution of the speakers.

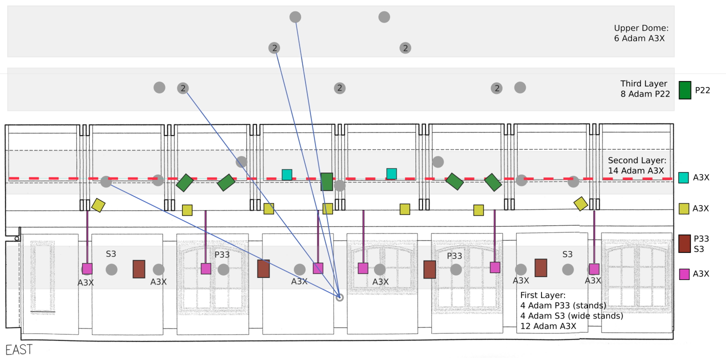

Figure 4: Side view of the Stage with speaker mounting points. Grey dots are ideal positions in a hemispherical dome, colored rectangles are the real positions

The final speaker configuration at which we arrived was an ear level ring of 20 speakers (the 8 original towers plus 12 additional A3X speakers), another ring of 14 A3X speakers mounted on the trusses (roughly 20 degrees in elevation above the first ring), and the original 8 speakers (roughly 20 degrees of elevation higher) plus 6 more A3X's distributed in the upper part of the dome. The 12 ear level speakers could not be mounted on stands that would take away floor space needed for seating, and had to be able to be moved out of the way when not in use. We installed a truss mounted rail system and designed telescoping mounts that could be switched between the normal listening position and a "parked" position where the 12 small speakers are moved next to the existing towers. The mechanical design took a long time and several prototypes were built and tested. Our final system features custom fabricated mounts made from 80/20 extruded aluminum profiles and hanging steel channel to facilitate rolling the speakers between locations.

5. DRIVING MANY SPEAKERS

One of the difficult aspects of the design process was finding an audio routing and distribution technology that would allow us to satisfy all the requirements within a reasonable amount of time and with the limited budget and manpower available to us. Furthermore, the full system needed to be controlled from a computer running GNU/Linux (like our Listening Room system), and Linux desktops and laptops should be able to connect to it for diffusion tasks.

For our GRAIL concert sound diffusion system we had been using a homebrew system which consisted of one half of a network snake (the Mamba box), plus some ingenious software in the form of a Jack[10] client (jack-mamba [11]), to transform it into a very reliable 32 channel D/A converter. While the system proved to be rock solid for our concerts, it was not really expandable in a way which could satisfy our requirements.

The first audio technology we explored was MADI. We had used RME MADI audio interfaces which had good driver support in Linux in our Listening Room system. For this 22.4 system we had to use two cards, one RME MADI and one RayDAT. This type of system could scale up to the number of inputs and outpus that we needed, but we could not find an easy way to control rerouting of connections to

Figure 5: Speaker mount

support both modes of operation. The only reasonable cost option we found was an RME MADI switching matrix, but switching between MADI scenes required several operations on the front control panel, and there was no option for remote software control which would have enabled us to design a separate simple to use interface.

Our experience with the ethernet based Mamba digital snake system suggested that a similar technology based on ethernet could be an answer to meet our requirements.

There are several protocols that rely on ethernet connections to transport audio and interconnect several audio interfaces together. The most widespread commercially so far has been Dante, but that was ruled out as the protocol specification is closed and proprietary, and there is no formal support for Linux. There is one company that offers a 128 channel ethernet card with associated Linux binary drivers, but there is no guarantee that this will be supported for future kernel upgrades and the card and driver combo is extremely expensive.

AVB (Audio Video Bridging) [12], on the other hand, is an open standard with a free software implementation embodied in the openAVNu project [13]. Regretfully not many manufacturers have used this standard for their products. One product manufacturer we considered was Motu, as their newer audio interfaces can be connected to each other through AVB and standard ethernet cables. Their internal configuration can be completely controlled through a built-in web server which makes it platform agnostic, and there is a published API that can use JSON http requests and OSC to remotely control all aspects of its operation. A Linux computer could control the full system without relying on proprietary software.

Regretfully the AVNu project does not yet include code for a complete Linux-based solution. It would be possible to create one, but that would require a substantial software development effort which was beyond the scope of the resources available to this project.

We bought a couple of interfaces for evaluation and experimented with using their USB interfaces. In the most desirable MOTU cards we found that the implementation of the USB2 class compliant driver was limited to 24 channels, which was much less than what we

needed (the cards were advertised as having 64 channel I/O through USB2, but that was only possible when using their proprietary driver). So we were at an impasse.

5.1. Firmware giveth...

Almost by chance we found an online reference to a "64 channel mode", and traced it back to a very recent firmware upgrade that added a mode selection configuration option to the USB audio interface. The new firmware allowed us to set the maximum number of channels handled by the USB class compliant driver to 64 if the sampling rate was limited to 44.1 and 48KHz, which was acceptable for our use case. This is beyond what the USB2 specification can do, but it performed well in tests under Linux, and allowed us to potentially address all speakers through the GRAIL control computer's USB2 interface, while multiple additional audio interfaces could communicate audio data through AVB. This new feature also would enable end users to interface with the finished diffusion system using another audio interface with its own USB2 interface. This would provide multiple entry points into the system using just USB2, making it easily usable by our users.

A firmware upgrade transformed the Motu hardware into a viable option. But what firmware can give, it can take away, as we will see...

5.2. Digital Mixer Mode

The first phase of the design centered around finding a configuration that could keep the old setup of DM1000 plus 8 main speakers operational with minimal changes. Some simple tests determined that routing the DM1000 to a 16A Motu interface through ADAT so it would drive the speakers (instead of the DM1000 driving them directly) would not change the latency of the system significantly. This 16A audio interface would also be the word clock master for the whole system, and this basic setup would depend on only the DM1000 and that interface being up and running to work.

This means that the 16.8 legacy system (we will call this the "Digital Mixer Mode") could be kept unchanged, and could be a subset of the full 48.8 system (the "OpenMixer Mode").

5.3. Routing the Subwoofers

There was a very long design detour that tried to use the internal crossover of the old subwoofers in "Digital Mixer Mode" as they were working fine and everybody wanted to keep their well known sound. We are going to skip those 4 months and jump straight into the design that incorporated new subwoofers much later.

The subwoofer upgrade proved to be a problem, both from the point of view of signal routing and from the specs that they had to meet. We wanted to have standalone crossovers when in "Digital Mixer Mode", and software crossovers implemented in the GRAIL control computer when in "OpenMixer Mode". We also wanted to have a rather high crossover frequency (originally 110Hz, currently about 90Hz) to minimize the cone excursion of the main speakers at low frequencies (they are mid-field monitors and almost too small for the space, but we love their very precise sound). And we wanted a low frequency limit of around 20Hz with enough power to fill the room without clipping or distortion.

The ideal subwoofer that would meet all our requirements does not exist (the details of why that is the case are beyond the scope of this paper). We ended up buying SVS SB4000 units, and not using the internal DSP processing included in the unit.

The only workable solution we found was to use external programmable crossovers when the system was operating in "Digital Mixer Mode". We used DBX 260 units and routed them through inputs and outputs of the same Motu audio interface used to drive the 8 main speakers (this back and forth tour added a tiny bit of latency). In "Digital Mixer Mode" the DBX crossovers are inserted into the signal path by the internal routing of the Motu audio interfaces, and in "OpenMixer" mode they are completely disconnected so that the GRAIL control computer can directly interface with speakers and subwoofers, and provide its own separate digital crossovers. In "Digital Mixer Mode" the signals going to the 8 main speakers are routed to the crossovers which split it between the main speakers and to the corresponding subwoofers, in "OpenMixer Mode" all speakers are mixed in to the 8 subwoofers. All the signal switching is accomplished using the routing matrix that is part of the Motu audio interfaces.

The use of external crossovers also allowed us to properly match phase at the crossover frequency and equalize the whole system in "Digital Mixer mode" for best performance, something we could not do before the upgrade.

Another 16A Motu interface drives the upper 8 speakers with signals that are sent from the digital mixer through AVB and the internal routing matrices of both audio interface cards.

The core system in "Digital Mixer Mode" consists of two Motu 16A cards, the DBX crossover units and the DM100 digital mixer. That not only keeps the same operational characteristics as before, but improves the system through better crossovers and speakers.

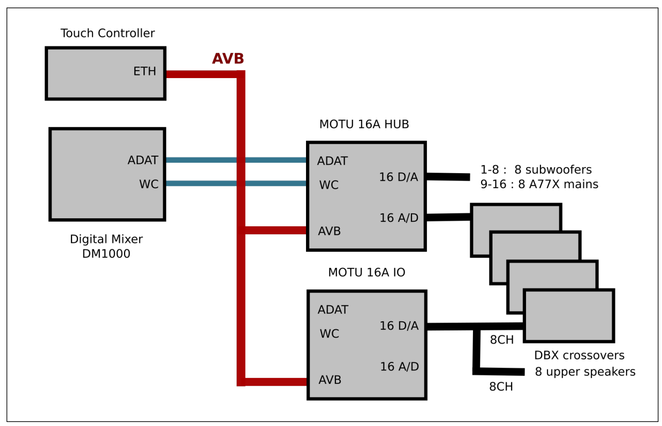

Figure 6: Signal routing in Digital Mixer Mode

5.4. And firmware taketh away...

In the middle of the design and implementation of the system we found that newer Motu interfaces no longer had the 64 channel mode configuration option. It turns out that Motu had "unspecified problems" with it, and removed the feature from their products through another firmware upgrade.

Suddenly the audio interfaces were useless for our purposes (24 channels instead of 64), with no fix coming from Motu, after all, they worked fine with their proprietary drivers. To make a long story short, we were able to downgrade the firmware to a version where that feature was still supported, and everything worked again. A not

very sustainable fix and a hack as we (and possibly random users of the system) have to ignore the constant reminders that a “software upgrade is available”.

While software upgradable products offer useful flexibility, you never know when something you depend on might go away, or some new and exciting capabilities might be added, and in which order that might happen. That was not the last problem we had with Motu firmware versions.

5.5. OpenMixer Mode

With the core architecture now a working reality, we added three 24Ao Motu audio interfaces hidden in the ceiling trusses of the Stage to drive the additional 32 small speakers (two would have been enough, but using three made the wiring easier and wiring represented a large time expenditure in this upgrade). An additional 16A in the OpenMixer control computer rack (on casters) acted as the interface between the OpenMixer Linux control computer and the rest of the system, using a single USB2 interface. AVB streams are used to send and receive audio to all other Motu audio interfaces, and finally to all speakers, and changing the internal routing in the audio interfaces through JSON http calls configures the audio routing for the two main modes of operation.

An additional 24Ai audio interface in the OpenMixer system rack is the entry point in the system for connecting laptops and other computers for concert diffusion or other purposes (Windows, OSX and Linux are all supported). A single USB2 cable allows us to have up to 64 channels of input/output available, which is enough for our current needs. AVB and the internal routing of the interfaces is used to send signals around.

Yet another 16A audio interface is used to interface with our dedicated Linux desktop workstation which resides on another cart together with its display, keyboard and mouse. A total of 8 Motu audio interfaces interconnected through AVB make up the audio part of the diffusion system.

Three Motu AVB switches connect all the audio interfaces together, and the different racks and mobile units in the space are easily connected through long ethernet cables (one mobile rack for the digital mixer and associated equipment, another for the OpenMixer control computer and another one for the desktop computer). The use of ethernet means there is a **significantly** smaller cable count to manage 64 channels of audio.

5.6. Switching modes

The attentive reader might have noticed that switching between “Digital Mixer Mode” and “OpenMixer Mode” seems to be happening magically so far. While we do have a Linux control computer, we cannot rely on it for switching modes. The system should keep working even if the control computer is off, or if it breaks down.

A solution that has worked admirably well is to add yet another computer (as if the system was not complex enough). This additional computer is a RaspberryPI 3 with a touch panel, mounted right next to the digital mixer. It allows the user to switch sampling rates, switch between operating modes and even activate different options in “OpenMixer mode” (changing between the Direct and Ambisonics modes, selecting Ambisonics decoders, etc). It communicates through ethernet with all the Motu audio interfaces and the main OpenMixer control computer.

The OpenMixer control computer also has a touch display, and

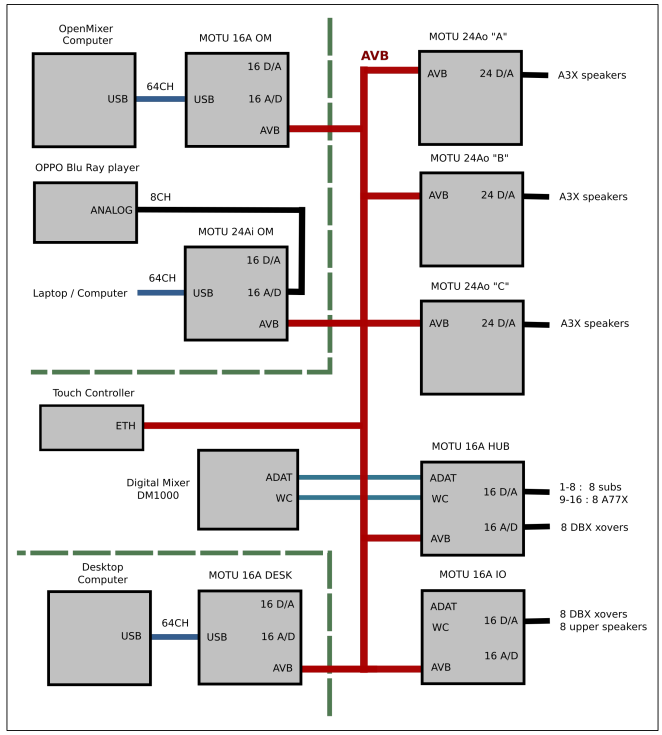

Figure 7: Signal routing in OpenMixer Mode

the software was designed so that either of them can be used to control the system and they stay synchronized with each other.

5.7. What? More Speakers?

Quite early in the implementation process Christopher Jette pushed for the immediate inclusion of something we had planned as a future expansion. In addition to the existing subwoofer and main speaker, the eight main towers would house 8 speakers almost hugging the ground. These speakers were included to help “pull down” the sound image, specially in the Ambisonics decoder modes. So our final speaker count is 56 speakers and 8 subwoofers, adding up to 64 individual outputs. We are maxed out.

5.8. Control Software

In “OpenMixer Mode” the Linux control computer (currently booting Fedora and running an optimized RT patched kernel) performs all internal DSP using SuperCollider[14] and its Supernova multi-core load-balancing sound server [15]. Jconvolver [16] is used for very efficient low latency partitioned convolution, and implements the digital loudspeaker correction filters. The software itself is conceptually simple, it provides for level and delay equalization of all speakers, digital crossovers (a combination of Linkiwitz Rayley [17] and Butterworth filters), routing control so that different sound sources (digital mixer, laptop, desktop) can be connected to the speakers, optional built-in Ambisonics decoders created with ADT [3][4](up to 6th order) and of course digital equalization of all speakers with convolution filters created from analyzing their measured impulse responses with the DRC (Digital Room Correction [18]) software package.

SuperCollider is started automatically on boot through a systemd

unit and takes care of orchestrating the rest of the system startup process. First, Jack [10] is started, then the SuperCollider program starts the Supernova sound server and its associated DSP software, two instances of Jconvolver, and finally everything is connected together using aj-snapshot and dynamically generated XML connection files. SuperCollider monitors all auxiliary programs, and restarts and reconnects them if they somehow fail.

The whole system is optimized for low latency, and currently runs with 128 frames per period (work is underway to get it to work at 64 frames per period, which would start approaching the performance of the digital mixer which runs with 64 frame blocks).

SuperCollider is also used for the touch graphical user interface in both the main computer and the small RaspberryPi switching appliance.

5.9. Calibration

For best performance the full speaker array is calibrated after the initial installation and when hardware changes are made. First elevation and azimuth angles for all speakers are measured, as well as the distances to the center of the space. These measurements are used to create the Ambisonics decoders for the main array and the subwoofers, and also to compensate for arrival times at the center of the space. After that we use Aliki [19] to measure the impulse response of the speakers, and that information is used to calculate convolution filters using DRC. Finally SPL measurements are done to compensate for small differences in speaker loudness in both direct and Ambisonics modes.

6. PROBLEMS AND CHALLENGES

While the selection of Motu products lead to a viable design, there are still occasional problems when using them on "unsupported platforms".

Occasionally an audio interface can disconnect from one or more of its AVB streams. The web interface shows them blinking and we have not found a way out of this other than rebooting both interfaces. After the reboot the connections are re-established automatically. We have not been able to find a way to reproduce this, and it only happens in the more complex Stage system we are describing in this paper (it has not happened, so far, in a far simpler system now running in our Listening Room). We have to do an thorough audit of the existing streams and only enable exactly what we need. This may be a problem solved in later firmware releases, but we are chained to older ones to retain the features that make the system possible in the first place.

In a different Studio in which we also deployed a single Motu interface we found another firmware related problem when using the class compliant driver under Linux. Suddenly inputs going into the computer through USB would switch channels in blocks of 8. What was coming through input 1 is suddenly in input 9, and so on and so forth. Again, downgrading to a previous firmware version fixes the problem (or using the proprietary driver). *Caveat emptor*.

In terms of the Linux control computer for the Stage system, the long term solution for interfacing with the audio interfaces is to use AVB streams directly. That would lift the 64 channel limitation (we of course would like to add a few more speakers), and hopefully make the system more reliable. The foundation of that is available in the OpenAVNu git repository but much work remains to be done (some preliminary tests managed to sync the Linux computer to the AVB clock, and get the system to recognize the existence of a Motu card).

6.1. Motu vs. Jack vs. PulseAudio

A weird feature of the Motu interfaces is that every time the sampling rate is changed (even if it is an internal change and the card is not slaved to an external clock) it takes the card a few seconds to acquire a "lock". During this time Jack can try to start, but at some point it decides that it can't, and fails.

This can lead to an endless loop of failed starts in the following scenario: assume the card is already running at 44.1KHz and we are trying to start Jack at 48KHz. Jack requests exclusive access to the card from PulseAudio and the request is granted. Jack tries to start but fails, because the card was running at 44.1KHz and it takes time to switch to 48KHz. After the attempt the card is switching to 48KHz, but when Jack quits it hands the card back to PulseAudio, which promptly resets its sampling rate to its default, 44.1KHz. And we are back where we started. There is no way to start Jack, unless PulseAudio is killed or its default sampling rate is changed to the one we want, or we tell it to ignore the card, which is not what we want to do.

If there is no change in sampling rate and Jack fails to start, waiting a few seconds and trying again succeeds.

To avoid this problem, in the control software for both the Listening Room and Stage Linux computers we use a JSON http call to check the lock status of the audio interface clock and delay the start of Jack until the sampling rate is locked.

7. CONCLUSIONS

The opening concerts of the newly upgraded Stage took place in October 4/5 2017, and the system performed very well (at the time we were still using the old subwoofers). Another round of upgrades in 2018 replaced the original subwoofers with newer ones, as outlined above, and also upgraded the main 8 speakers with newer A77X Adam monitors. The lower layer of speakers were repositioned at the bottom of the main towers, and the new subwoofers were stacked immediately above them (originally they had been reversed). A second round of successful concerts (our annual Transitions concerts) took place in October 2018 with the fully upgraded array. The full array has seen more use in the past year, with several concerts using it instead of what would have been stereo or quad diffusion.

We have outlined the design process of a complex Linux-based diffusion system, using off-the-shelf components and GNU/Linux for all the software components.

8. ACKNOWLEDGMENTS

This would have been impossible to accomplish without the support of CCRMA and its community. Many many hours of discussions made for a better system that satisfies all the use cases of the space. Endless critical listening tests honed the system into better and better sound quality. Many thanks to Eoin Callery for his contributions to the design and keen ears, and for keeping us grounded at all times (we tend to fly away). The whole project would not have happened had we not had Christopher Jette on the CCRMA Staff at the time. He pushed and worked and talked and discussed and designed and kept things going. Invaluable. Matt Wright, our Technical Director, also spent many hours helping with big and small details. Many students helped, in particular thanks to Megan Jurek,

Figure 8: Transitions 2018 concert

who spent many hours soldering many many small connectors, and routing what seemed like miles of cables. No audio would flow if not for her help. Jay Kadis, our audio engineer at the time, also spent quite a bit of time wiring DB25 connectors and cabling the main towers. Juan Sierra, one of our MA/MST students, was instrumental in properly phase matching of the new subwoofers with the main speakers and tuning the crossovers for best performance, the Stage sounds much better thanks to him. Carlos Sanchez, sysadmin and staff at CCRMA, designed and implemented the hardware and software that drives the touch interface that controls the whole system. And Constantin Basica, our new concert coordinator, has been helping visiting artists use the full system for much more interesting concerts over the past year. Many thanks to all involved, we can now do justice to many fantastic pieces from composers that tickle our ears with beautiful sounds arranged in space.

9. REFERENCES

[1] Fernando Lopez-Lezcano, “Searching for the grail,” *Computer Music Journal*, vol. 40, no. 4, pp. 91–103, 2016.

[2] Franz Zotter and Matthias Frank, “All-round ambisonic panning and decoding,” *J. Audio Eng. Soc*, vol. 60, no. 10, pp. 807–820, 2012.

[3] Aaron Heller, Eric Benjamin, and Richard Lee, “A toolkit for the design of ambisonic decoders,” *Proceedings of the Linux Audio Conference 2012*, 2012.

[4] Aaron Heller and Eric Benjamin, “The ambisonics decoder toolbox: Extensions for partialcoverage loudspeaker arrays,” *Proceedings of the Linux Audio Conference 2014*, 2014.

[5] Thibaut Carpentier, Natasha Barrett, Rama Gottfried, and Markus Noisternig, “Holophonic sound in ircam’s concert hall: Technological and aesthetic practices,” *Computer Music Journal*, vol. 40, no. 4, pp. 14–34, 2016.

[6] Fernando Lopez-Lezcano and Jason Sadural, “Openmixer: a routing mixer for multichannel studios,” in *Proceedings of the Linux Audio Conference 2010*, 2010.

[7] Elliot Kermit-Canfield and Fernando Lopez-Lezcano, “An update on the development of openmixer,” *Proceedings of the Linux Audio Conference 2015*, 2015.

[8] A. B. J. Kuijlaars E. B. Saff, “Distributing many points in a sphere,” in *The Mathematical Intelligencer*, Volume 19, Number 1, 1997.

[9] “Openscad, the programmers solid 3d cad modeller,” `http://www.openscad.org/`.

[10] Paul Davis, “Jack audio connection kit,” `http://jackaudio.org/`, 2002.

[11] Fernando Lopez-Lezcano, “From Jack to UDP packets to sound and back,” in *Proceedings of the Linux Audio Conference 2012*, 2012.

[12] “The avnu alliance (avb),” `https://avnu.org/`.

[13] “Openavnu git repository,” `https://github.com/AVnu/OpenAvnu`.

[14] J. McCartney, “Supercollider: A new real-time synthesis language,” in *Proceedings of the International Computer Music Conference*, 1996.

[15] Tim Blechmann, “Supernova: a multiprocessor aware real-time audio synthesis engine for supercollider,” M.S. thesis, TU Wien, 2011.

[16] Fons Adriaensen, “Jconvolver, a convolution engine,” `http://kokkinizita.linuxaudio.org/linuxaudio/`, 2006.

[17] Siegfried Linkwitz, “Active crossover networks for noncoincident drivers,” in *Journal of the Audio Engineering Society*, Volume 24 Issue 1, 1976, pp. 2–8.

[18] Denis Sbragion, “Drc: Digital room correction,” `http://drc-fir.sourceforge.net/`, 2002.

[19] Fons Adriaensen, “Aliki, an integrated system for impulse response measurements,” `http://kokkinizita.linuxaudio.org/linuxaudio/`, 2006.

[20] Ingo Molnar and Thomas Gleixner, “Real-time linux, the preempt_rt patches,” `https://wiki.linuxfoundation.org/realtime/start`, 2000.

Figure 9: Transitions 2017 concert